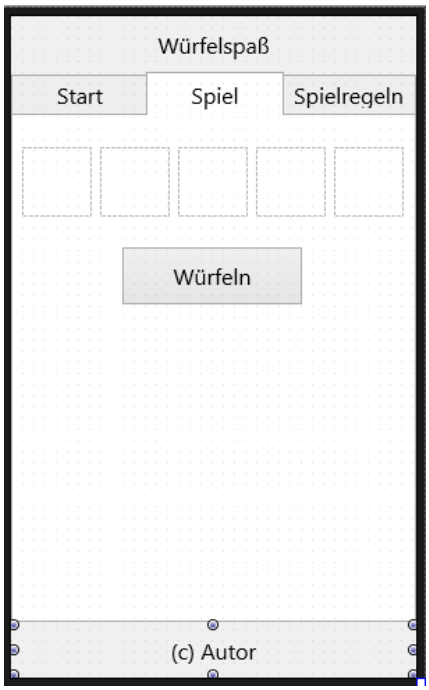
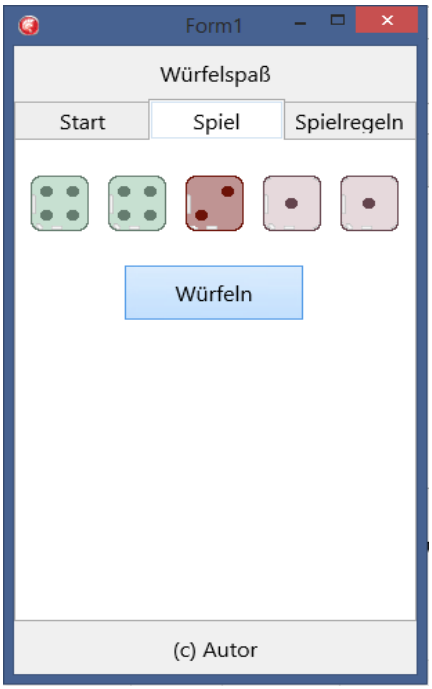
Informatik Würfeln in einer App

1. Öffne ein neues Projekt in Delphi XE7. Datei-> neu-> geräteübergreifende Anwendung. Wähle “Leere Anwendung“. Ein leeres Fenster erscheint. Links unten im Objektinspektor siehst du einige Eigenschaften des Fensters und kannst sie ändern. Wähle im Objektinspektor Height = 500 (567) und Width = 300 (384). Speichere das Projekt ab unter dem Namen „Wurf“ und die Unit (Pascal-Datei) unter „Wurf1“. Die Endungen werden automatisch angehängt.
2. Wähle rechts unten in der Tool-Palette unter „Standard“ ein TToolbar-Objekt und lege es auf dem Formular ab. Da die Voreinstellung bei diesem Objekt align = top ist, wird es sofort oben im Formular angeordnet. Gegebenenfalls wird der Name von ToolBar1 auf TopToolBar geändert. Lege auf dem Toolbar-Objekt ab ein Label-Objekt ab. Setze StyleLookup = toolbuttonlabel, align = client, TextSettings.HorzAlign = Center und Text auf den Namen des Würfelspiels. Wähle nun ein zweites TToolbar-Objekt und lege es auf dem Formular ab. Da die Voreinstellung bei diesem Objekt align = top ist, ändere align auf bottom und es wird sofort unten im Formular angeordnet. Gegebenenfalls wird der Name von ToolBar1 auf BottomToolBar geändert. Lege auf dem Toolbar-Objekt ab ein Label-Objekt ab. Setze StyleLookup = toolbuttonlabel, align = client, TextSettings.HorzAlign = Center und Text auf den Namen Programmierers – eventuell mit dem Copyright-Zeichen.
3. Wähle nun im Eintrag „Common Controls Elemente“ das Objekt tTabControl und lege es unterhalb der Toolbar im sogenannten Clientbereich des Formulars ab. Setze align = client. Dann klicke mit der rechten Maustaste auf das Objekt und füge mit dem Eintrags-Editor drei Karteikarten-Seiten, im folgenden nur noch Seiten genannt, hinzu. Wähle für dieses TabControl1 im Objektinspektor FullSize = true. Klicke nacheinander auf „TabItem1“ bis „TabItem3“ und setze Text nacheinander auf „Start“, „Spiel“ und „Spielregeln“. Setze ebenfalls bei jedem TabItem TextSettings -> Font -> Schriftgrad auf 16.
4. Gehe nun auf die Seite „Spiel“. Lege dort am rechten Rand ein tImage-Objekt ab. Entwirf nun mit einem Bildeditor ein 50x50 Punkte großes Bild eines Würfels, der die Zahl Eins anzeigt. Verwende dabei außen die Farbe Weiß. Speichere das Bild unter Bild1.png im Verzeichnis der Anwendung. Lade es nun wie folgt in das Image-Objekt: Kliche auf Image1 und im Objektinspektor auf MultiResBitmap. Nun öffnet sich ein Dialogfenster. Setze dort „Transparente Farbe“ auf weiß. Öffne sodann die Bilddatei „Bild1“. Es erscheint nun das Bild im Dialogfenster mit einem gepunkteten Muster außen. Drücke nun den Schalter mit dem grünen Häkchen (Ok) und das Image-Objekt enthält das Bild mit transparentem Aussehen am Rand. Verfahre so für alle 6 Würfelergebnisse. Jetzt (oder auch später) wird bei allen Image-Objekten visible= false gesetzt, da sie nicht die eigentlichen Würfel sind, sondern nur die Wurfvorlagen. Beim Würfeln werden diese Wurfvorlagen in die eigentlichen Würfel-Image-Objekte kopiert, wie es im nächsten Punkt beschrieben wird.
5. Nun werden wir das Würfeln anschaulich darstellen. Dazu erstellen wir die eigentlichen Spielwürfel, hier 5 an der Zahl. Wir legen oben auf der Seite „Spiel“ nebeneinander 5 Image-Objekte ab und benennen sie mit „W1“ bis „W5“. Ebenfalls legen wir auf der Seite „Spiel“ ein tButton-Objekt ab und beschriften es mit „Würfeln“. Die folgenden Bilder zeigen Entwurf und Ablauf.





In der OnClick-Methode dieses Schalters schreiben wir

```
procedure TForm1.Button1Click(Sender: TObject);
var WurfErgebnis,Spalte:integer;  Startzeit,Stoppzeit:tDateTime;
begin
   Startzeit:= Time;
   repeat
      for Spalte:=1 to 5 do begin
         if tImage(FindComponent('W'+IntToStr(Spalte))).Opacity=1 then begin
            WurfErgebnis:= random(6)+1;
            tImage(FindComponent('W'+IntToStr(Spalte))).MultiResBitmap:=
               tImage(FindComponent('Image'+IntToStr(WurfErgebnis))).MultiResBitmap;
            WEFeld[Spalte]:= WurfErgebnis;
         end;
      end;
      sleep(30);
      Application.ProcessMessages;
   until SecondOf(Time-Startzeit)>1;
end;
```

Die Funktion `SecondOf()` ist in der Unit System.DateUtils deklariert, die oben unter „uses“ hinzugenommen werden muss. Weiter oben im Programmtext unter der Deklaration der Variablen Form1 wird das Array für das Wurfergebnis gespeichert, um es auch rechnerisch auszuwerten.

```
var
  Form1: TForm1;
  WEFeld : array[1..5] of integer;
```

6. Es soll nun mit denjenigen Würfeln nicht weiter gewürfelt werden, die markiert wurden. Zum Markieren setzen wir die Durchlässigkeit herab. Diese Würfel sehen dann inaktiv aus. Dazu schreiben wir in der OnClick-Methode des Image-Objektes W1:

```
procedure TForm1.W1Click(Sender: TObject);
begin
   if tImage(Sender).Opacity = 1 then tImage(Sender).Opacity:=0.3
                                 else tImage(Sender).Opacity:=1
end;
```

Diese Methode von W1 wird jetzt im Objektinspektor für W2 bis W5 als OnClick-Methode ausgewählt. Achtung: Sie wird nicht neu geschrieben!!